NUWC-NPT Technical Report 11,129
26 July 1996

Efficient Calculation of Directivity Indices for Certain Three-Dimensional Arrays

Albert H. Nuttall
Surface Undersea Warfare Directorate

Benjamin A. Cray
Submarine Sonar Department

Naval Undersea Warfare Center Division
Newport, Rhode Island

PREFACE

This technical report was prepared by Dr. Benjamin A. Cray for the Office of Naval Research (ONR), Submarine Technology Branch under the Conformal Acoustic VElocity Sensor (CAVES) program. The CAVES program is funded by the ONR Science and Technology Directorate, Program Manager, Dr. Vernon P. Simmons (Code 334). The primary type of funding is 6.3A; NUWC assignment no. SS0102, NUWC project no. A19840.

Some of the work described here was co-sponsored by the Independent Research (IR) Program of NUWC, Project B10007, entitled *Near-Optimum Detection of Random Signals With Unknown Locations, Structure, Extent, and Strengths*. The IR program is funded by the Office of Naval Research; the NUWC Division Newport program manager is Dr. Stuart C. Dickinson (Code 102). The principal investigator for this IR project is Dr. Albert H. Nuttall (Code 311).

The technical reviewer for this document was Dr. Howard H. Schloemer (Code 2002).

REVIEWED AND APPROVED: 26 July 1996

R. J. Martin
Head, Submarine Sonar Department

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	26 July 1996	Final

4. TITLE AND SUBTITLE

Efficient Calculation of Directivity Indices for Certain Three-Dimensional Arrays

5. FUNDING NUMBERS

B10007
A19840

6. AUTHOR(S)

A. H. Nuttall and B. A. Cray

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Undersea Warfare Center
Detachment New London
39 Smith Street
New London, Connecticut 06320-5594

8. PERFORMING ORGANIZATION REPORT NUMBER

TR 11,129

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

Array directivity is defined as the ratio of the output signal-to-noise ratio of an array to the input signal-to-noise ratio at an omni-directional element in an isotropic noise field. Calculation of directivity is obtained by integrating the magnitude-squared response of the array over all angles of incidence. In spherical coordinates, these arrival angles are denoted by an azimuthal angle, θ, and a polar angle, ϕ. Hence, calculation of the directivity requires a two-fold integration over the angular space defined by the azimuthal and polar angles. Except for a few simple array geometries and unique frequencies, analytical solutions for this two-fold integration can generally not be obtained. Numerical integration techniques must therefore be used to calculate directivity. For large-aperture arrays, consisting of thousands of array elements, conventional integration procedures, such as a double application of Simpson's rule over azimuth and elevation angles, is woefully time consuming and can be potentially inaccurate. This study presents a number of procedures which dramatically decrease the time required to numerically calculate directivity for certain array shapes and weights. Algorithms, written on a SUN SPARC station10 in Fortran 77, are provided in the appendices. However, the factorization procedures described here can be implemented equally well in other programming languages, such as MATLAB.™ The enclosed programs, which incorporate all of the procedures described herein, can be used to calculate the directivity of certain planar arrays, as well as three-dimensional arrays which are conformal to the cylindrical portion of a submarine hull, for example.

14. SUBJECT TERMS		15. NUMBER OF PAGES
Directivity	Factorization	43
Arrays	Directional Noise	16. PRICE CODE
Conformal Array		

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	SAR

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

EFFICIENT CALCULATION OF DIRECTIVITY INDICES FOR CERTAIN THREE-DIMENSIONAL ARRAYS

INTRODUCTION

Directivity Index (DI) may be defined as a measure of the improvement in the signal-to-noise power ratio (S/N) that a beamformed array provides in an *ideal isotropic noise field* with a perfectly correlated signal, relative to an omni-directional array element in the free-field, i.e.,

$$DI = 10 \log DF = 10 \, Log \frac{(S/N)_{array\ output}}{(S/N)_{free-field\ omni-element}}. \qquad (1)$$

For an array with arbitrary geometry and element locations, the beamformed array single-frequency amplitude response[1] may be written as

$$A(\theta,\phi) = \sum_{t=1}^{T} w_t \; g_t(\theta,\phi) \; \exp(i(\vec{k} - \vec{k}_s) \cdot \vec{r}_t), \qquad (2)$$

where $A(\theta,\phi)$ is the array's angular response to an arrival from azimuthal angle θ and polar angle ϕ, T is the total number of array elements, w_t and g_t are, respectively, the amplitude shading and element angular sensitivity of the t-th array element, $\vec{k}$ is the wavevector corresponding to an arriving acoustic planewave, and $\vec{k}_s$ is the wavevector to which the array is steered. The corresponding steering angles are θ_s, ϕ_s. The t-th array element position is defined by the coordinate vector $\vec{r}_t$.

A spatial coordinate system (with corresponding wavevectors) for an array oriented in the xz-plane is illustrated in figure 1, along with the definition of spherical angles θ,ϕ. The programs supplied in the appendices of this report, to calculate directivity, use the geometry of figure 1 to define input parameters. For the programs **planar-xz-equal.f** and **planar-xz-grid.f**, a rectangular array in the xz-plane is assumed. Program **planar-xz-grid.f** allows for arbitrary element grid spacing, that is, d_x or d_z may vary from element-to-element. The program **volumetric-grid.f** assumes an array configuration which allows for cylindrical curvature about the x-axis. At each fixed array row x_ℓ, elements are located at the same positions $\{y_n, z_n\}$; if all y_n are equal, the array becomes a plane parallel to the xz-plane.

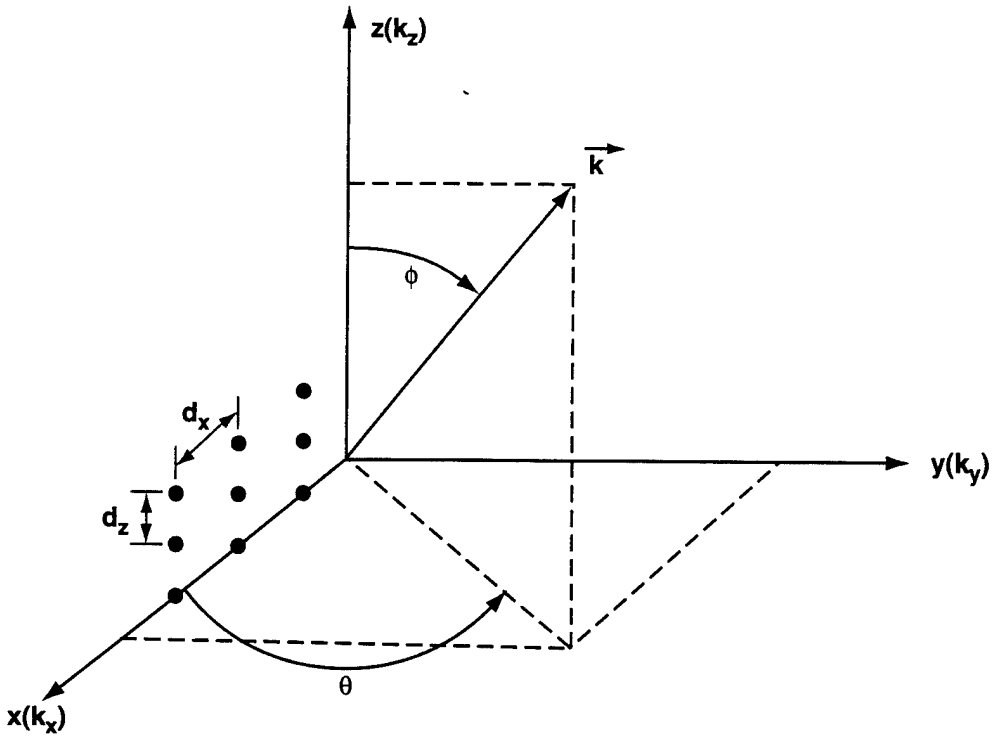

Figure 1. Array Coordinate System and Spherical Angles Denoting Incident Planewave Arrival Direction θ,ϕ

With the aid of equations (1) and (2), the components which comprise the calculation of the directivity index can be written as

$$S_{ff-omni} = S, \tag{3a}$$

where S is defined as the given signal power level, while

$$S_{array} = S\left|A(\theta_s,\phi_s)\right|^2, \tag{3b}$$

$$N_{ff-omni} = \int_0^{2\pi}\int_0^{\pi} N\sin(\phi)d\phi\, d\theta\,, \tag{3c}$$

where N is defined as the isotropic noise level, while

$$N_{array} = \int_0^{2\pi} \int_0^{\pi} N\left|A(\theta,\phi)\right|^2 \sin(\phi)\, d\phi d\theta \,. \tag{3d}$$

Noting that equation (3c), is simply $4\pi N$, and canceling the signal and isotropic noise levels, equation (1) reduces to

$$DI = 10\, Log\left\{ \frac{4\pi\left|A(\theta_s,\phi_s)\right|^2}{\int_0^{2\pi}\int_0^{\pi}\left|A(\theta,\phi)\right|^2 \sin(\phi)\, d\phi d\theta} \right\}, \tag{4}$$

where $\left|A(\theta_s,\phi_s)\right|^2$ is the array power response at steering angles θ_s,ϕ_s.

Equation (3d) indicates an integral with limits that account for all acoustic arrival angles. Hence, the directivity expression above assumes an unbaffled, or free-field, array configuration. In a baffle configuration, N_{array} would be reduced, resulting in an increased directivity. However, the programs supplied here include directional (such as cosine to a power) element sensitivities, $g_t(\theta,\phi)$ in (2), which inherently provide the gains realized from a baffle. There is no need to correct the directivity values yielded by the programs to account for baffling of isotropic free-field noise.

For a line array with constant inter-element spacing, the double integral in equation (4) may be evaluated exactly at the frequency for which spacing $d_x = \lambda/2$, regardless of the steering direction. This leads to

$$DI = 10 \log\left\{ \frac{\left(\sum_{t=1}^{T} w_t\right)^2}{\sum_{t=1}^{T} w_t^2} \right\}. \tag{5}$$

Accurate directivity predictions for arbitrary array configurations, at general frequencies of interest, require numerically integrating the double integral given in equation (4) *and* calculating the maximum array power response, $\left|A(\theta_m,\phi_m)\right|^2$. It should be noted that the maximum array response may not occur precisely at the steering angles (θ_s,ϕ_s). Array

curvature, unequal array element spacing, element sensitivity, and shading weights act to shift the maximum response axis from (θ_s,ϕ_s) to (θ_m,ϕ_m). The programs supplied here have global and local search procedures which obtain the peak power response and the corresponding angles (θ_m,ϕ_m).

The remainder of this report will discuss accurate and efficient numerical integration methods for evaluating the double integral of equation (4). The focus is to reduce the number of evaluations required of the integrand, in order to achieve rapid convergence and accuracy with a minimum of computational effort.

For very small partitions, the Riemann sum will provide a reasonable approximation of the definite integrals in equation (4). Indeed, if an exceedingly large number of narrow rectangles (or augmented partitions) are used in the Riemann sum, the approximation would be quite accurate. This implies, however, an enormous amount of function evaluations, each of which is computationally time-consuming. For a large number of array elements, on the order of thousands, such a simplistic approach would be unfeasible. Vector and matrix operations used in high-performance computation software, such as MATLAB®, do provide quick function evaluations and have built-in numerical integration routines. However, these computational benefits can be reduced by the amount of RAM available on a given computer system. Swapping memory space between a system's hard drive and RAM can be inefficient; resulting in lengthy integration times. The goal then is to use an integration method which yields a highly accurate integral approximation with a minimum number of function evaluations.

Due to the periodicity and analyticity of the integrand $|A(\theta,\phi)|^2$ in azimuthal angle θ, an elementary closed-type Trapezoidal formula was used for evaluating the outer θ integral in equation (4) over a full period $(0,2\pi)$. The inner integral, over polar angle ϕ, uses a Gauss-Legendre quadrature formula of order m, where m can vary over the twelve values (16, 24, 32, 48, 64, 96, 128, 192, 256, 384, 512, 768). The Gauss-Legendre method is generally significantly more accurate than an equal-interval formula, such as the Trapezoidal rule, for a given number of function evaluations. However, Gauss quadrature was deliberately not chosen for the outer θ-integral due to the particular integration limits and the integrand periodicity and analyticity in θ. The Trapezoidal rule is extremely efficient for integration of an analytic periodic integrand, when conducted over a full period.[2] Generally, efficient numerical evaluation of multi-dimensional integrals requires examination of the nature of the integrand and the integration limits; rarely is one procedure optimum for all integrals.

FACTORIZATION OF ARRAY RESPONSE

The array amplitude response $A(\theta,\phi)$ was given in equation (2) for the general three-dimensional array with arbitrary element locations, weights, and element responses, where T is the total number of elements in the array. In the general case, the sum of T terms can be evaluated only by calculating every individual complex term and summing up those T quantities. This is a particularly time-consuming task, especially when T is of the order of 10,000 or more. When coupled with the fact that the directivity index requires computation of a double integral, namely equation (4), which must be repeated for each different frequency and/or steering angle, the computational burden becomes excessive. Furthermore, as the number T of elements increases, the array power response $|A(\theta,\phi)|^2$ becomes even sharper in angles θ,ϕ, thereby requiring still finer evaluation of the integrand in equation (4), in order to retain accuracy in the final DI calculation.

In this section, we will significantly reduce this computational burden for a particular class of arrays and weights, by factoring the amplitude response $A(\theta,\phi)$ into a product of two sums which have far fewer total terms than T. For example, an array of 50 by 200 elements will require evaluation of 50 + 200 = 250 terms instead of 50 * 200 = 10,000 terms, a savings in execution time by a factor of 40. Additional savings in time are achieved by precomputing and storing quantities that are used repeatedly in the program for calculation of the DI. This storage requirement is minimal, even for the very large arrays of interest here.

ELEMENT LOCATIONS

We use the coordinate system illustrated in figure 1. Receiving elements will be located only on L planes of fixed x, namely, at $\{x_\ell\}$ for $1 \le \ell \le L$, where each x_ℓ is arbitrary. Thus, these planes need not be equally spaced in x.

At each x-coordinate x_ℓ, N elements are then located at the points y_n,z_n for $1 \le n \le N$; these pairs $\{y_n,z_n\}$ are identical for all x_ℓ. However, each location y_n,z_n is arbitrary. This generality allows for the array to take on a cylindrical shape, for example; other more general three-dimensional arrays are also allowed. In the special case where the $\{y_n\}$ are all equal, this becomes a planar array, parallel to the xz-plane; however, the L locations $\{x_\ell\}$ and the N locations $\{z_n\}$ are still arbitrary.

The array is seen to contain a total of T = L N elements, in L parallel planes of N elements each, with an identical repeated (arbitrary) configuration in each x-plane. This is the general configuration considered here. Any three-dimensional array which cannot be put into this form will not be covered by the following analysis and simplifications.

Each element at x_ℓ, y_n, z_n has its normal in the yz-plane, at an angle ψ_n relative to the z-axis; see figure 2. That is, the element lies in the yz-plane for every element in the array; however, the element orientation angle ψ_n can vary with location y_n, z_n, but not with location x_ℓ. Thus, the direction vector of the n-th element boresight is $0\ \mathbf{i} + \sin(\psi_n)\ \mathbf{j} + \cos(\psi_n)\ \mathbf{k}$ for the element at x_ℓ, y_n, z_n.

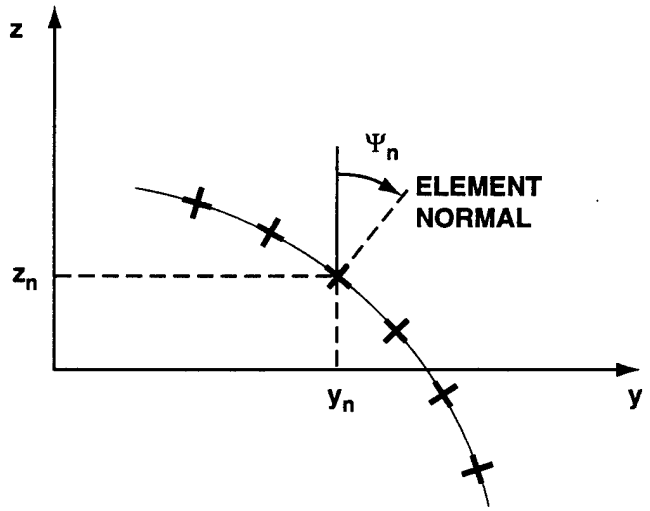

Figure 2. Element Locations at a Fixed x_ℓ

ELEMENT RESPONSES

The cosine of the angle $B_n(\theta,\phi)$ between the n-th element normal and a general arrival direction θ,ϕ is given by the dot product

$$\begin{aligned}\cos B_n(\theta,\phi) &= [0\,\mathbf{i} + \sin(\psi_n)\,\mathbf{j} + \cos(\psi_n)\,\mathbf{k}] \bullet \\ &\bullet [\cos(\theta)\sin(\phi)\,\mathbf{i} + \sin(\theta)\sin(\phi)\,\mathbf{j} + \cos(\phi)\,\mathbf{k}] = \\ &= \sin(\psi_n)\sin(\theta)\sin(\phi) + \cos(\psi_n)\cos(\phi)\,.\end{aligned} \tag{6}$$

Notice that this quantity is independent of ℓ and x_ℓ, due to the element locations and boresight orientations.

We are interested here in relative element responses of the form

$$r_n(\theta,\phi) = [\cos B_n(\theta,\phi)]^{\upsilon} \quad \text{for } |B_n(\theta,\phi)| \le \pi/2\,, \tag{7}$$

where υ is a power under our control. By means of the procedure presented in Appendix A, the element responses can be approximated by

$$r_n(\theta,\phi) \cong \exp\left[-\upsilon\left(u_n + \frac{1}{2}u_n^2 + \frac{1}{3}u_n^3 + \cdots + \frac{1}{I}u_n^I\right)\right], \tag{8}$$

where I is an integer under our control, and

$$u_n \equiv 1 - \cos B_n(\theta,\phi) = 1 - \sin(\psi_n)\sin(\theta)\sin(\phi) - \cos(\psi_n)\cos(\phi)\,. \tag{9}$$

Here, we also used equation (6). We will use the element response $r_n(\theta,\phi)$ to an arrival from direction θ,ϕ, as given by equation (8) and (9), for all the calculations herein. The region of applicability is $0 \le \theta \le 2\pi$, $0 \le \phi \le \pi$.

ELEMENT WEIGHTS

A multiplicative weight $\alpha_n \beta_\ell$ is applied at general element location x_ℓ, y_n, z_n for $1 \le \ell \le L$, $1 \le n \le N$. The total effect of weighting and element response is therefore given by

$$w_{n\ell}(\theta,\phi) = \alpha_n \, \beta_\ell \, r_n(\theta,\phi) . \tag{10}$$

It is important to observe that this total effective element response is factorable in the variables n and ℓ; it need not be factorable in θ and ϕ.

ARRAY AMPLITUDE RESPONSE

For notational simplicity, the following definitions will be used below:

$$\underline{x}_\ell \equiv 2\pi x_\ell / \lambda , \quad \underline{y}_n \equiv 2\pi y_n / \lambda , \quad \underline{z}_n \equiv 2\pi z_n / \lambda , \quad \exp i(x) \equiv \exp(ix) , \tag{11}$$

where λ is the wavelength of the single-frequency arriving planewave. When the features above are incorporated into the array amplitude response (2), it can be rewritten in the form

$$\begin{aligned} A(\theta,\phi) = \sum_{n=1}^{N} \sum_{\ell=1}^{L} w_{n\ell}(\theta,\phi) \quad \exp i\Big[& \underline{x}_\ell \cos\theta \sin\phi + \underline{y}_n \sin\theta \sin\phi + \underline{z}_n \cos\phi \\ & - \left(\underline{x}_\ell \cos\theta_s \sin\phi_s + \underline{y}_n \sin\theta_s \sin\phi_s + \underline{z}_n \cos\phi_s \right) \Big] , \end{aligned} \tag{12}$$

where θ_s, ϕ_s is the steering direction, and θ, ϕ is the planewave arrival direction. Finally, when the particular form of weighting in (10) is employed, amplitude response (12) further simplifies to the desired factored form

$$\begin{aligned} A(\theta,\phi) = & \sum_{n=1}^{N} \alpha_n \, r_n(\theta,\phi) \exp i\left(\underline{y}_n \sin\theta \sin\phi + \underline{z}_n \cos\phi - a_n \right) \times \\ & \times \sum_{\ell=1}^{L} \beta_\ell \exp i\left(\underline{x}_\ell \cos\theta \sin\phi - b_\ell \right) , \end{aligned} \tag{13}$$

where parameters

$$a_n \equiv \underline{y}_n \sin\theta_s \sin\phi_s + \underline{z}_n \cos\phi_s\,, \quad b_\ell \equiv \underline{x}_\ell \cos\theta_s \sin\phi_s\,. \tag{14}$$

The computation of amplitude response $A(\theta,\phi)$ by means of (13) requires an N-fold summation on n and a L-fold summation on ℓ, which constitutes a total of N + L complex terms. When these complex operations are completed, the magnitude-squared operation yielding power response $|A(\theta,\phi)|^2$ is taken. The complex function expi(x) ≡ exp(ix) = cos(x) + i sin(x).

NUMERICAL EVALUATION OF DOUBLE INTEGRAL

The major computation required in DI calculation (4) is the double integral

$$V \equiv \int_0^{\pi} d\phi \sin\phi \int_0^{2\pi} d\theta \; |A(\theta,\phi)|^2\,. \tag{15}$$

When the factorized expression (13) is substituted in (15), the result is expressible in the form

$$V = \int_0^{\pi} d\phi \sin\phi \int_0^{2\pi} d\theta \left| \sum_{\ell=1}^{L} \beta_\ell \, \mathrm{expi}\left(\underline{x}_\ell \cos\theta \sin\phi - b_\ell\right) \right|^2 \times$$
$$\times \left| \sum_{n=1}^{N} \alpha_n \, r_n(\theta,\phi) \, \mathrm{expi}\left(\underline{y}_n \sin\theta \sin\phi + \underline{z}_n \cos\phi - a_n\right) \right|^2 . \tag{16}$$

Element response $r_n(\theta,\phi)$ is given by (8) and (9).

The integrand in (16) is analytic in θ and φ. Furthermore, it has period 2π in θ, and the θ integral is over this complete interval. This makes the θ integral in (16) a good candidate for numerical evaluation via the trapezoidal rule [2; pp. 53-59]. On the other hand, the φ integral is only over (0, π), which is half the period in φ. This suggests the Gauss-Legendre rule for numerical integration on φ. This mixed procedure has been utilized for the approximate evaluation of (16).

The specific formula for M-point Gauss-Legendre integration of function f(x) over interval (a,b) has the general form

$$\int_a^b dx\, f(x) = \frac{b-a}{2}\int_{-1}^{1} dt\, f\left(\frac{b+a}{2}+\frac{b\text{-}a}{2}\,t\right) \equiv \frac{b\text{-}a}{2}\sum_{m=1}^{M} w_m\, f\left(\frac{b+a}{2}+\frac{b\text{-}a}{2}\,t_m\right) =$$

$$= \frac{b\text{-}a}{2}\sum_{m=1}^{M} w_m\, f\left(x_m\right); \quad x_m \equiv \frac{b+a}{2}+\frac{b\text{-}a}{2}\,t_m\,. \tag{17}$$

The quantities $\{t_m\}$ are the M zero locations of the M-th order Legendre polynomial $P_M(x)$, and are symmetrically located in the (-1,1) interval. The M Gauss weights $\{w_m\}$ are all positive. These quantities (t_m and w_m for M up to 768) are required as input to the numerical programs provided in the appendices. Gauss-Legendre weights and zero-locations are available in many references.[3] When combined with the trapezoidal rule for integration on θ, there follows for (16), with $\phi_m \equiv \frac{1}{2}\pi + \frac{1}{2}\pi\, t_m$, $\theta_k = 2\pi k/K$,

$$V \equiv \frac{\pi}{2}\sum_{m=1}^{M} w_m \sin\phi_m \,\frac{2\pi}{K}\sum_{k=1}^{K}\left|\sum_{\ell=1}^{L}\beta_\ell \exp i\left(\underline{x}_\ell \cos\theta_k \sin\phi_m - b_\ell\right)\right|^2 \times$$

$$\times\left|\sum_{n=1}^{N}\alpha_n\, r_n\left(\theta_k,\phi_m\right)\exp i\left(\underline{y}_n \sin\theta_k \sin\phi_m + \underline{z}_n \cos\phi_m - a_n\right)\right|^2, \tag{18}$$

where the element response is now expressible as

$$r_n(\theta_k,\phi_m) = \exp\left[-\upsilon\left(u+\frac{1}{2}u^2+\frac{1}{3}u^3+\cdots+\frac{1}{I}u^I\right)\right] \tag{19}$$

with

$$u \equiv 1 - \sin\psi_n \sin\theta_k \sin\phi_m - \cos\psi_n \cos\phi_m\,. \tag{20}$$

For trapezoidal integration, the interval $(0,2\pi)$ was cut into K panels, each of width $2\pi/K$. However, the two usual edge weights of 1/2 have been combined into a single unity weight at $\theta = 2\pi$, by making use of the periodicity in θ. Hence, the summation on k in (18) goes from 1 to K, using unity weights throughout.

Expression (18) is the final result to be used for numerical evaluation of the double integral for V in (15), when the array is three-dimensional. Quantities such as $\sin\theta_k$, $\sin\phi_m$, etc., that can be done once and for all, have been precomputed and stored for access in (18) as required, in order to minimize the execution time. A program for the evaluation of (18) is available in Appendix B under the title **volumetric-grid.f**.

For a nonisotropic noise field which depends only on polar angle ϕ, say with power dependence $D(\phi)$, the only changes required are to insert $D(\phi)$ after $\sin\phi$ in (15), and to insert $D(\phi_m)$ after $\sin\phi_m$ in (18). Of course, the integral of product $\sin(\phi)\,D(\phi)$ over $(0,\pi)$ must also be evaluated.

SPECIALIZATION TO PLANAR ARRAY, ARBITRARY GRID

When all the y-coordinates $\{y_n\}$ are zero (or constant), the array reduces to a planar array parallel to the xz-plane; however, the element locations can still be irregular, namely at each intersection x_ℓ, z_n for $1 \le \ell \le L$, $1 \le n \le N$. Every x-coordinate x_ℓ is arbitrary, and every z-coordinate z_n is arbitrary in this grid structure.

When the boresight of every individual element is perpendicular to the array plane, angle ψ_n in figure 2 is equal to $\pi/2$ for all n. This makes $u = 1 - \sin\theta_k \sin\phi_m$ in (20), which is independent of n; therefore, $r_n(\theta_k,\phi_m)$ in (19) is also independent of n, allowing it to be factored out of the n summation in (18).

Also, when all the $\{y_n\}$ are zero, the latter expi term in (18) becomes independent of k, allowing the entire magnitude-squared quantity in the second line of (18) to be factored out of the k summation. The end result is the simplified form

$$V \equiv \frac{\pi}{2} \sum_{m=1}^{M} w_m \sin\phi_m \left| \sum_{n=1}^{N} \alpha_n \mathrm{expi}\left(z_n \cos\phi_m - a_n\right) \right|^2 \times$$

$$\times \frac{2\pi}{K} \sum_{k=1}^{K} \left| r(\theta_k,\phi_m) \right|^2 \left| \sum_{\ell=1}^{L} \beta_\ell \mathrm{expi}\left(x_\ell \cos\theta_k \sin\phi_m - b_\ell\right) \right|^2 , \qquad (21)$$

where now

$$\left|r(\theta_k,\phi_m)\right|^2 = \exp\left[-2\left(u+\frac{1}{2}u^2+\frac{1}{3}u^3+\ldots+\frac{1}{I}u^I\right)\right], \qquad (22)$$

with

$$u = 1-\sin\theta_k \sin\phi_m\,. \qquad (23)$$

It has been found that these manipulations and simplifications result in approximately 30 percent savings in computation time relative to the more general volumetric form (18). A program for simplified form (21) is listed in Appendix C under the title planar-xz-grid.f, where the xz qualifier emphasizes the fact that the planar array is parallel to the xz-plane.

SPECIALIZATION TO PLANAR ARRAY, EQUAL SPACINGS

A useful additional simplification is possible when the planar array has equally spaced elements in both the x- and z-coordinates. Specifically, suppose that the element locations are given by

$$x_\ell = \ell\, d_x \quad \text{for } 1\le \ell \le L\,, \quad z_n = n\, d_z \quad \text{for } 1\le n\le N\,. \qquad (24)$$

Substitution of these values into (21), and reference to (11) and (14), results in the modified form

$$V \equiv \frac{\pi}{2}\sum_{m=1}^{M} w_m \sin\phi_m \left|\sum_{n=1}^{N}\alpha_n \mathrm{expi}\left(\frac{2\pi}{\lambda}d_z(\cos\phi_m-\cos\phi_s)n\right)\right|^2 \frac{2\pi}{K}\times$$
$$\sum_{k=1}^{K}\left|r(\theta_k,\phi_m)\right|^2\left|\sum_{\ell=1}^{L}\beta_\ell \mathrm{expi}\left(\frac{2\pi}{\lambda}d_x(\cos\theta_k\sin\phi_m-\cos\theta_s\sin\phi_s)\ell\right)\right|^2 \qquad (25)$$

where (22) and (23) are still relevant.

The reduction in computation time is now obtained by taking advantage of the recursive capability of the expi function:

$$\mathrm{expi}[n\gamma] = \exp[in\gamma] = \exp[i\gamma]\exp[i(n-1)\gamma] = \mathrm{expi}[\gamma]\mathrm{expi}[(n-1)\gamma]\,. \qquad (26)$$

Thus, one complex multiplication gives the next term necessary in the n or ℓ summations in (25), without the need for any storage.

It has been found that these manipulations and simplifications result in approximately 60 percent additional savings in computation time relative to the more general planar form (21). A program for simplified planar form (25) is listed in Appendix D under the title planar-xz-equal.f, where the xz-qualifier again emphasizes the fact that the planar array is parallel to the xz-plane.

RESULTS

Directivity calculations, using a low-order quadrature formula, are given in Ref. [4] for a 9-element line array, a 60-element planar array and a 21-element cylindrical array. For arrays with relatively few elements, such as these, there is little need to be concerned with optimal integration procedures. Computationally efficient integration routines can provide order-of-magnitude speed up times, but this is of little importance if the original calculation only takes a few seconds. However, the arrays considered here are large, with dense element spacings, and the frequency and steering angular dependence of the directivity must be examined in detail. Therefore, it is essential to have an efficient means to accurately evaluate the two-fold integral of equation (4).

Table 1 below summarizes the planar and conformal array lay-outs used for preliminary numerical directivity calculations. Directivity was calculated at a number of frequencies and assumed a sound speed of 4,900 ft/sec.

Table 1. Planar and Volumetric Array Input Parameters for Directivity Calculations

Parameter	Input Field	**planar-xz equal.f**	**volumetric-grid.f**
No. of Elements along x-axis	L = 200	√	√
No. of Elements along z-axis	N = 50	√	**X**
Element spacing along x-axis	Δ_x = 3.5 in.	√	√
Element spacing along z-axis	Δ_z = 3.6 in.	√	**X**
Element spacing (arc length)	d_s = 3.6 in.	**X**	√
θ_s-steering angle (azimuthal)	$\theta_s = \pi/2$ rads	√	√
ϕ_s-steering angle (polar)	$\phi_s = \pi/2$ rads	√	√
θ-angle increment	$\Delta\theta$ = 0.0001 rads	√	√
ϕ-angle increment	$\Delta\phi$ = 0.0001 rads	√	√

X = Input not required.

For both examples, the total number of array elements was 10,000. The angular increment parameters shown are necessary in order to obtain a reasonable estimate of the maximum array response, $|A(\theta_m,\phi_m)|^2$. In general, the number of angular samples is related to the array length, and frequency, i.e., we must require angular sampling of the order

$$(\Delta\theta, \Delta\phi) \leq \frac{\lambda}{4L_e}, \tag{27}$$

where λ is the acoustic wavelength and $L_e = \left(L_x^2 + L_y^2\right)^{\frac{1}{2}}$. For the numerical programs provided here, this θ and ϕ sampling requirement corresponds to the restriction $K > 2\pi L_e/\lambda$ for θ–sampling, where K is the program parameter kc, and $M > \frac{\pi^2}{2} L_e/\lambda$ for ϕ-sampling, where M is the program parameter mc. Additional input parameters, common to each program, are given in Appendix B.

The computed directivity is given in table 2. As expected, the planar array geometry yields slightly greater directivity than that of the conformal array; the differences are due to array curvature. With factorization, it took 154 seconds to compute the directivity value 38.57 dB (at 4,000 Hz) using the program volumetric-grid.f on a SUN SPARC Station 10. Without factorization, the time required to compute this value was on the order of 1 hour, 10 minutes.

Table 2. Directivity DI (dB) Calculated From planar-xz-equal.f and volumetric-grid.f

Frequency (Hz)	DI planar-xz-equal.f (dB)	DI volumetric-grid.f (dB)
4,000	**38.71**	**38.57**
5,000	**40.63**	**40.49**
6,000	**42.21**	**42.06**
7,000	**43.54**	**43.39**
8,000	**44.69**	**44.54**

As previously discussed, array gains in nonisotropic noise fields (having variations in polar angle only) can be evaluated using the programs supplied here. In many ocean environments, such as in the Bering or Norwegian Seas, a distinctive and well-defined notch or dipole-like directivity pattern is present in ambient noise. This variation has been described as a vertical noise profile[5] and may be modeled using the continuous expression

$$D(\phi) = 1 - A_d \exp\left\{-\frac{(\phi - \phi_d)^2}{2\sigma_d^2}\right\}. \tag{28}$$

Figure 3 illustrates a typical nonisotropic noise power dependence for the parameters $A_d = 0.9$, which controls the notch depth, $\phi_d = \pi/2$, the notch location, and $\sigma_d = 0.1$, the notch width.

Table 3 provides the array gain (AG) calculated from planar-xz-equal.f and volumetric-grid.f assuming the above vertical noise power distribution. The gain over isotropic noise varies from 6.8 dB to 7.9 dB; certainly, the vertical noise profile has a significant effect on AG calculations.

Table 3. Array Gain (AG) for Nonisotropic Noise Calculated From the planar-xz-equal.f and volumetric-grid.f Programs

Frequency (Hz)	DI planar-xz-equal.f (dB)	DI volumetric-grid.f (dB)
4,000	45.54	45.32
5,000	47.87	47.65
6,000	49.73	49.52
7,000	51.28	51.07
8,000	52.60	52.38

APPROXIMATION FOR DIRECTIVITY INDEX

The approximation $10 \log(4\pi A/\lambda^2)$ for the directivity index DI of an equi-spaced baffled planar array is only applicable for an unweighted array with half-omni directionality and an isotropic noise field. However, the restriction to an unweighted array can be eliminated if the area A is replaced by the product of effective lengths in the x- and z-directions. That is, we have, more generally, for a planar array with spacings d_x and d_z,

$$DI \equiv 10 \log\left(\frac{4\pi E_x E_z}{\lambda^2}\right), \tag{29}$$

where effective lengths E_x and E_z are given, respectively, by

$$E_x \equiv d_x \frac{\left(\sum_{\ell=1}^{L} \beta_\ell\right)^2}{\sum_{\ell=1}^{L} \beta_\ell^2}, \quad E_z \equiv d_z \frac{\left(\sum_{n=1}^{N} \alpha_n\right)^2}{\sum_{n=1}^{N} \alpha_n^2}. \tag{30}$$

For the isotropic noise example above, at 4,000 Hz, but now with Hanning weighting in the x-direction and flat weighting in the z-direction, the DI as found from the approximation is 33.88 dB, whereas the exact value turns out to be 33.85 dB, a discrepancy of only 0.03 dB. And for Hanning weighting in both directions, the approximate DI is 32.12 dB, whereas the exact value is 32.00 dB, a difference of 0.12 dB.

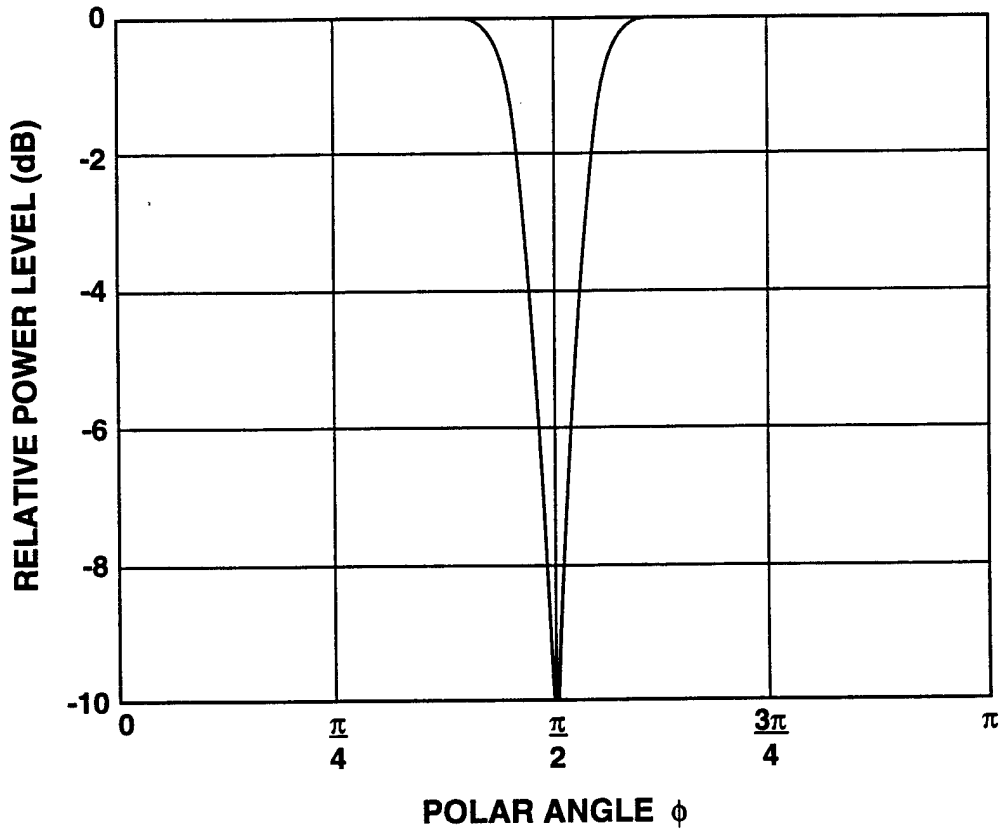

Figure 3. Typical Variation of Vertical Noise Directionality Assuming Noise Power Dependence D (ϕ) With $A_d = 0.9$, $\phi_d = \pi/2$, and $\sigma_d = 0.1$

REFERENCES

1. B. A. Cray, "Sonar Subsystem Lecture for the Submarine Combat Course," Massachusetts Institute of Technology Summer Professional Program, NUWC-NPT Technical Memorandum 951054, Naval Undersea Warfare Center Detachment, New London, CT, 12 June 1995

2. P. J. Davis and P. Rabinowitz, *Numerical Integration,* Blaisdell Publishing Company, Waltham, MA, 1967.

3. M. Abramowitz and I. E. Stegun, *Handbook of Mathematical Functions With Formulas, Graphs, and Mathematical Tables*, National Bureau of Standards, U.S. GPO, 1972.

4. D. Lee, "Computation of Beam Patterns and Directivity Indices for Three-Dimensional Arrays with Arbitrary Element Spacings," NUSC Technical Report 4687, Naval Underwater Systems Center, New London, CT, 22 February 1974.

5. T. Revello, "Beamformed Environment Noise," Presentation to the Naval Undersea Warfare Center Detachment, New London, CT, 11 December 1995.

APPENDIX A — APPROXIMATION TO THE ELEMENT RESPONSE

The power response of a baffled directive receiving element can often be fairly well approximated by cos(x) for $|x| < \pi/2$, where x is the angle between the element boresight angle and the planewave arrival angle. For $|x| > \pi/2$, the response is ideally zero. The discontinuity in slope of the response at $x = \pm\pi/2$ creates a problem for any numerical integration technique which does not specifically account for the locations of these two singularities.

In this appendix, we will derive an analytic approximation to this truncated cosine, where the degree of fit can be as tight as desired. Actually, since in practice, the truncated cos(x) response itself is not exact, an approximation to it is quite satisfactory and acceptable. It will immediately be obvious how the method extends to a truncated version of $\cos^{\upsilon}(x)$, where power υ is arbitrary.

We begin by observing that

$$\cos(x) = \exp[\ln\cos(x)] \quad \text{for } |x| < \pi/2 . \tag{A-1}$$

Alternatively,

$$\cos(x) = \exp[\ln(1-\{1-\cos(x)\})] = \exp[\ln(1-u)] , \tag{A-2}$$

where

$$u \equiv 1 - \cos(x) . \tag{A-3}$$

But, from the expansion

$$\ln(1-u) = -\left(u + \frac{1}{2}u^2 + \frac{1}{3}u^3 + \cdots\right) \quad \text{for } |u| < 1 , \tag{A-4}$$

that is, $|x| < \pi/2$, we immediately obtain

$$\cos(x) = \exp\left[-\left(u + \frac{1}{2}u^2 + \frac{1}{3}u^3 + \cdots\right)\right] \quad \text{for } |x| < \frac{\pi}{2}, \tag{A-5}$$

where u = 1 - cos(x) from (A-3).

Now, suppose we terminate the infinite series in (A-5) at I terms, and use the approximation (where T{ } denotes truncated)

$$T\{\cos(x)\} \cong \exp\left[-\left(u + \frac{1}{2}u^2 + \frac{1}{3}u^3 + \cdots + \frac{1}{I}u^I\right)\right] \tag{A-6}$$

for <u>all</u> x in the extended range $[-\pi,\pi]$, with u = 1 - cos(x). For large I, the right-hand side of (A-6) is a good approximation to cos(x) for $|x| < \pi/2$. And, for $|x| \geq \pi/2$, where $u \geq 1$, the u series would tend to ∞ as $I \rightarrow \infty$, meaning that the right-hand side of (A-6) is approximating zero, as desired. Furthermore, the right-hand side of (A-6) is always positive for all x, it has period 2π in x, and it is smooth for all x. A sample plot of the left-hand side and right-hand side of (A-6) for I = 5 is shown in figure A-1. The transition in behavior near $x = \pm\pi/2$ is now smooth.

The approximation in (A-6) uses cos(x) directly, namely, in the form u = 1 - cos(x). But, cos(x) can be interpreted as the direction cosine between the element boresight angle and the angle of arrival. Thus, any other element response, that can be written directly in terms of this quantity, can thereby use (A-6).

For example, for a power of cos(x), there follows from (A-6),

$$T\{\cos^{\upsilon}(x)\} \cong \exp\left[-\upsilon\left(u + \frac{1}{2}u^2 + \frac{1}{3}u^3 + \cdots + \frac{1}{I}u^I\right)\right] \tag{A-7}$$

for $|x| \in [-\pi,\pi]$, with u = 1 - cos(x).

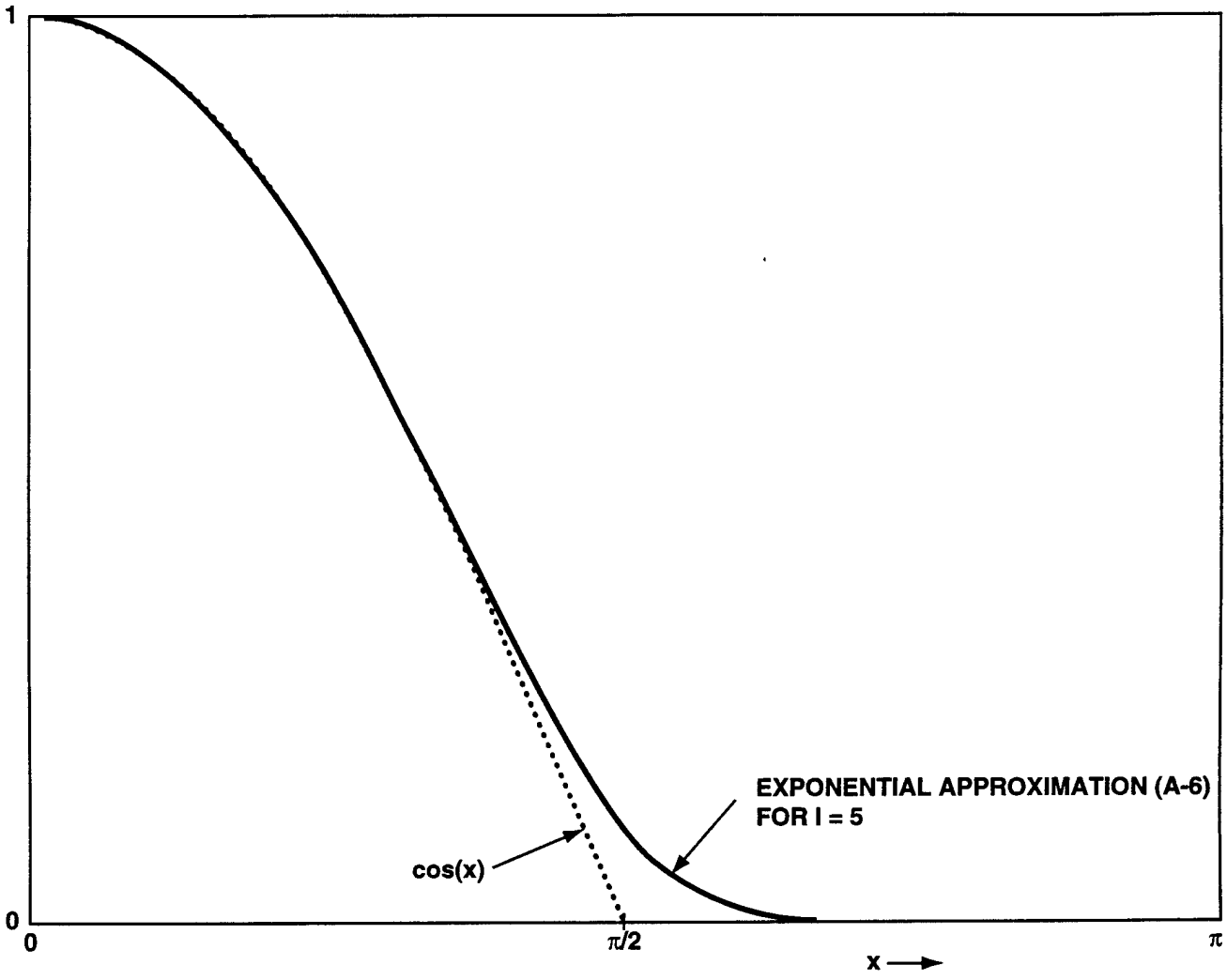

Figure A-1. Sample Plot of Equation A-6

APPENDIX B — Program volumetric-grid.f

This program allows for arbitrary element locations $\{x_\ell\}$ for $1 \le \ell \le L$ and $\{z_n\}$, $\{y_n\}$ for $1 \le n \le N$. However, the particular example programmed utilizes equal spacing dx in the x-direction and equal arc lengths ds in the polar angle ϕ, for simplicity. Both of these assumptions can easily be generalized to fit the users cases. Then, the two lines assigning values to dx and ds would be eliminated.

Coding for Hamming array weighting has not been included in this and the following appendix, since these array programs can utilize unequal spacings in the x- and/or z-directions. It is certainly possible to include nonuniform weighting, provided that the desired shading function is properly projected onto the actual unequal element locations, accounting for their density in space.

COMMON INPUT PARAMETERS

K = 400, number of azimuthal θ samples over $(0,2\pi)$
M = 192, number of polar ϕ samples over $(0,\pi)$
f = 4,000 Hz
c = 4,900 ft/sec
$\theta_S = \pi/2$, $\phi_S = \pi/2$; broadside steering
$\Delta_X = 3.5$ inches, $\Delta_Z = 3.6$ inches for planar array
$\Delta_X = 3.5$ inches, $\Delta_S = 3.6$ inches for volumetric array

PROGRAM SYMBOL	MATH SYMBOL AND EXPLANATION
lc	L, number of elements in x-coordinate
nc	N, number of elements in z-coordinate
kc	K, number of θ samples over $(0,2\pi)$
mc	M, number of ϕ samples over $(0,\pi)$
ng	total number of gauss locations and weights
wx	storage of L array weights for x-coordinate
wz	storage of N array weights for z-coordinate
phis	ϕ_S, polar steering angle (radians)
thetas	θ_S, azimuthal steering angle (radians)
dphi	Δ_ϕ, phi increment in search for maximum
dtheta	Δ_θ, theta increment in search for maximum
j	gauss case number, $1 \le j \le 12$

```
      implicit real*8(a-h,o-z)        !  volumetric-grid.f
      real sec,dtime,time(1:2)
      parameter(lc=200,nc=50,kc=400,ng=2520) !  kc = number of theta samples
      dimension x(1:lc),y(1:nc),z(1:nc),psi(1:nc)
      dimension wx(1:lc),wyz(1:nc),ax(1:lc),ayz(1:nc)
      dimension xx(1:ng),ww(1:ng),c(1:kc),s(1:kc),cn(1:nc),sn(1:nc)
      dimension mm(0:12)/0,16,24,32,48,64,96,128,192,256,384,512,768/
1     format(d28.18) ! case 1  2  3  4  5  6   7   8   9  10  11  12 <-- j
2     format(2d25.16)
3     format("sec",4e15.5)
      pi=4.d0*atan(1.d0)

      sec=dtime(time)                 !  use < legendre-array-dp
      read 1, xx(1:ng)                !  gauss-legendre locations
      read 1, ww(1:ng)                !  gauss-legendre weights
      sec=dtime(time)
      print 3, sec,time
      print 2

      ampd=.9d0                       !  relative power dip, .le. 1
      phid=pi*.5d0                    !  polar angle of dip, radians
      sigd=.1d0                       !  width of dip, radians

      freq=4000.d0                    !  frequency, hertz
      speed=4900.d0*12.d0             !  sound speed, inches/second
      dx=3.5d0                        !  x spacing, inches
      ds=3.6d0                        !  arc length spacing, inches
      rc=200.d0                       !  radius of array, inches
      a=0.d0                          !  lower limit on phi, radians
      b=pi                            !  upper limit on phi, radians
      phis=pi*.5d0                    !  polar steering angle, radians
      thetas=pi*.5d0                  !  azimuth steering angle, radians
      dphi=.0001d0                    !  phi increment in search, radians
      dtheta=.0001d0                  !  theta increment in search, radians
      j1=8                            !  starting gauss case, j1 .ge. 1
      j2=10                           !  ending   gauss case, j2 .le. 12

      wavelength=speed/freq
      cx=2.d0*pi/wavelength*dx
      cyz=2.d0*pi/wavelength*rc

      w1=0.d0
      do n=1,lc
      x(n)=cx*n                       !  x element locations (arbitrary)
      wx(n)=1.d0                      !  x element weights (arbitrary)
      w1=w1+wx(n)
      end do

      w2=0.d0
      t=(nc+1)*.5d0
      da=ds/rc                        !  element angular spacing, radians
      do n=1,nc
      an=da*(t-n)
      y(n)=cyz*cos(an)                !  y element locations (arbitrary)
      z(n)=cyz*sin(an)                !  z element locations (arbitrary)
      psi(n)=pi*.5d0-an               !  element polar angles in y,z plane
      wyz(n)=1.d0                     !  yz element weights (arbitrary)
      w2=w2+wyz(n)
      end do
      w3=w1*w1*w2*w2

      t=sin(phis)
      u=cos(thetas)*t
      do n=1,lc
      ax(n)=x(n)*u
      end do
```

```
t=sin(thetas)*t
u=cos(phis)
do n=1,nc
ayz(n)=y(n)*t+z(n)*u
end do

dt=2.d0*pi/kc                    !  theta sampling increment
do k=1,kc
t=k*dt
c(k)=cos(t)
s(k)=sin(t)
end do

do n=1,nc
t=psi(n)
cn(n)=cos(t)
sn(n)=sin(t)
end do

ba1=(b+a)*.5d0
ba2=(b-a)*.5d0
m22=0
big=0.d0
t3=1.d0/3.d0

do j=1,12
mc=mm(j)                         !  mc = number of phi samples
m22=m22+mm(j-1)
if (j .lt. j1) go to 9
if (j .gt. j2) go to 9
v=0.d0
two=0.d0

do m=1,mc
wm=ww(m+m22)
phim=ba1+ba2*xx(m+m22)
sm=sin(phim)
cm=cos(phim)
dm=(phim-phid)/sigd
dm=1.d0-ampd*exp(-.5d0*dm*dm)    !  noise power distribution
vk=0.d0
two=two+wm*sm*dm

do k=1,kc
cs=c(k)*sm
ss=s(k)*sm
ar=0.d0
ai=0.d0
br=0.d0
bi=0.d0

do n=1,lc
wn=wx(n)
t=x(n)*cs-ax(n)
ar=ar+wn*cos(t)
ai=ai+wn*sin(t)
end do !n

do n=1,nc
u=1.d0-sn(n)*ss-cn(n)*cm         !  element response:
rn=exp(-u*(1.d0+u*(.5d0+u*(t3+u*(.25d0+u*.2d0)))))
wn=wyz(n)*rn
t=y(n)*ss+z(n)*cm-ayz(n)
br=br+wn*cos(t)
bi=bi+wn*sin(t)
end do !n
```

```
        ab=(ar*ar+ai*ai)*(br*br+bi*bi)
        vk=vk+ab
        if (ab .lt. big) go to 10
        big=ab
        phib=phim
        thetab=k
10      end do !k

        v=v+wm*sm*dm*vk
        end do !m

        v=v*ba2*dt                      !  trapezoidal and gauss rules
        two=two*ba2
        dfw=two*2.d0*pi*w3/v
        diw=10.d0*log10(dfw)
        print 21, v
21      format("double integral",d25.16)
        print 4, dfw,diw
4       format("dfw, diw",2d25.16)
        dfb=two*2.d0*pi*big/v           !  directivity factor
        dib=10.d0*log10(dfb)            !  directivity index
        print 5, dfb,dib
5       format("dfb, dib",2d25.16)

        sec=dtime(time)
        print 3, sec,time
9       end do !j

        thetab=thetab*dt
        print 2
        print 6, big,w3                 !  big = maximum power response
6       format("big, w3",2d25.16)
        print 7, phib,thetab            !  angles after coarse search
7       format("phib, thetab",2d25.16)

12      phio=phib                       !  fine search for maximum,
        thetao=thetab                   !  starting from phib,thetab
        do i=-1,1
        phi=phio+dphi*i
        cm=cos(phi)
        sm=sin(phi)
        do j=-1,1
        theta=thetao+dtheta*j
        cs=cos(theta)*sm
        ss=sin(theta)*sm
        ar=0.d0
        ai=0.d0
        br=0.d0
        bi=0.d0

        do n=1,lc
        wn=wx(n)
        t=x(n)*cs-ax(n)
        ar=ar+wn*cos(t)
        ai=ai+wn*sin(t)
        end do !n

        do n=1,nc
        u=1.d0-sn(n)*ss-cn(n)*cm        !  element response:
        rn=exp(-u*(1.d0+u*(.5d0+u*(t3+u*(.25d0+u*.2d0)))))
        wn=wyz(n)*rn
        t=y(n)*ss+z(n)*cm-ayz(n)
        br=br+wn*cos(t)
        bi=bi+wn*sin(t)
        end do !n
```

```
        ab=(ar*ar+ai*ai)*(br*br+bi*bi)
        if (ab .le. big) go to 11
        big=ab
        phib=phi
        thetab=theta
11      end do !j
        end do !i

        if (abs(phib-phio)+abs(thetab-thetao) .gt. 0.d0) go to 12
        print 2
        print 8, big
8       format("big",2d25.16)
        print 7, phib,thetab            !  angles after fine search
        dfb=two*2.d0*pi*big/v           !  directivity factor
        dib=10.d0*log10(dfb)            !  directivity index
        print 5, dfb,dib

        phib=phis                       !  fine search for maximum,
        thetab=thetas                   !  starting from phis,thetas
        big=0.d0
        k=0
14      phio=phib
        thetao=thetab
        do i=-1,1
        phi=phio+dphi*i
        if (k .eq. 0) phi=phis
        cm=cos(phi)
        sm=sin(phi)
        do j=-1,1
        theta=thetao+dtheta*j
        if (k .eq. 0) theta=thetas
        cs=cos(theta)*sm
        ss=sin(theta)*sm
        ar=0.d0
        ai=0.d0
        br=0.d0
        bi=0.d0

        do n=1,lc
        wn=wx(n)
        t=x(n)*cs-ax(n)
        ar=ar+wn*cos(t)
        ai=ai+wn*sin(t)
        end do !n

        do n=1,nc
        u=1.d0-sn(n)*ss-cn(n)*cm        !  element response:
        rn=exp(-u*(1.d0+u*(.5d0+u*(t3+u*(.25d0+u*.2d0)))))
        wn=wyz(n)*rn
        t=y(n)*ss+z(n)*cm-ayz(n)
        br=br+wn*cos(t)
        bi=bi+wn*sin(t)
        end do !n

        ab=(ar*ar+ai*ai)*(br*br+bi*bi)
        if (ab .le. big) go to 13
        big=ab
        phib=phi
        thetab=theta
        if (k .eq. 0) go to 16
13      end do !j
        end do !i
16      k=k+1
        if (k .eq. 1) go to 14
```

```
      if (abs(phib-phio)+abs(thetab-thetao) .gt. 0.d0) go to 14
      print 2
      print 8, big
      print 7, phib,thetab              !  angles near steering direction
      dfs=two*2.d0*pi*big/v             !  directivity factor
      dis=10.d0*log10(dfs)              !  directivity index
      print 15, dfs,dis
15    format("dfs, dis",2d25.16)

      end
```

APPENDIX C — Program planar-xz-grid.f

This program allows for arbitrary element locations $\{ x_\ell \}$ for $1 \le \ell \le L$ and $\{z_n\}$ for $1 \le n \le N$. However, the particular example programmed utilizes two equal spacings dx and dz, for simplicity; these assumptions can easily be generalized to fit the users cases. Then, the two input lines assigning values to dx and dz would be eliminated.

```
      implicit real*8(a-h,o-z)        !  planar-xz-grid.f
      real sec,dtime,time(1:2)
      parameter(lc=200,nc=50,kc=400,ng=2520) !  kc = number of theta samples
      dimension x(1:lc),z(1:nc),wx(1:lc),wz(1:nc),ax(1:lc),az(1:nc)
      dimension xx(1:ng),ww(1:ng),c(1:kc),s(1:kc)
      dimension mm(0:12)/0,16,24,32,48,64,96,128,192,256,384,512,768/
1     format(d28.18) ! case 1  2  3  4  5  6   7   8   9  10  11  12 <-- j
2     format(2d25.16)
3     format("sec",4e15.5)
      pi=4.d0*atan(1.d0)

      sec=dtime(time)                 !  use < legendre-array-dp
      read 1, xx(1:ng)                !  gauss-legendre locations
      read 1, ww(1:ng)                !  gauss-legendre weights
      sec=dtime(time)
      print 3, sec,time
      print 2

      ampd=.9d0                       !  relative power dip, .le. 1
      phid=pi*.5d0                    !  polar angle of dip, radians
      sigd=.1d0                       !  width of dip, radians

      freq=4000.d0                    !  frequency, hertz
      speed=4900.d0*12.d0             !  sound speed, inches/second
      dx=3.5d0                        !  x spacing, inches
      dz=3.6d0                        !  z spacing, inches
      a=0.d0                          !  lower limit on phi, radians
      b=pi                            !  upper limit on phi, radians
      phis=pi*.5d0                    !  polar steering angle, radians
      thetas=pi*.5d0                  !  azimuth steering angle, radians
      dphi=.0001d0                    !  phi increment in search, radians
      dtheta=.0001d0                  !  theta increment in search, radians
      j1=8                            !  starting gauss case, j1 .ge. 1
      j2=10                           !  ending   gauss case, j2 .le. 12

      wavelength=speed/freq
      cx=2.d0*pi/wavelength*dx
      cz=2.d0*pi/wavelength*dz

      w1=0.d0
      do n=1,lc
      x(n)=cx*n                       !  x element locations (arbitrary)
      wx(n)=1.d0                      !  x element weights (arbitrary)
      w1=w1+wx(n)
      end do

      w2=0.d0
      do n=1,nc
      z(n)=cz*n                       !  z element locations (arbitrary)
      wz(n)=1.d0                      !  z element weights (arbitrary)
      w2=w2+wz(n)
      end do
      w3=w1*w1*w2*w2

      u=cos(thetas)*sin(phis)
      do n=1,lc
      ax(n)=x(n)*u
      end do
      u=cos(phis)
      do n=1,nc
      az(n)=z(n)*u
      end do

      dt=2.d0*pi/kc                   !  theta sampling increment
      do k=1,kc
      t=k*dt                          !  theta(k)
```

```
      c(k)=cos(t)
      s(k)=sin(t)
      end do

      ba1=(b+a)*.5d0
      ba2=(b-a)*.5d0
      m22=0
      big=0.d0
      t3=1.d0/3.d0

      do j=1,12
      mc=mm(j)                          !  mc = number of phi samples
      m22=m22+mm(j-1)
      if (j .lt. j1) go to 9
      if (j .gt. j2) go to 9
      v=0.d0
      two=0.d0

      do m=1,mc
      wm=ww(m+m22)
      phim=ba1+ba2*xx(m+m22)
      sm=sin(phim)
      cm=cos(phim)
      dm=(phim-phid)/sigd
      dm=1.d0-ampd*exp(-.5d0*dm*dm)     !  noise power distribution
      vk=0.d0
      two=two+wm*sm*dm
      br=0.d0
      bi=0.d0

      do n=1,nc
      wn=wz(n)
      t=z(n)*cm-az(n)
      br=br+wn*cos(t)
      bi=bi+wn*sin(t)
      end do !n
      bsq=br*br+bi*bi

      do k=1,kc
      cs=c(k)*sm
      ss=s(k)*sm
      u=1.d0-ss                         !  element response:
      rsq=exp(-2.d0*u*(1.d0+u*(.5d0+u*(t3+u*(.25d0+u*.2d0)))))
      ar=0.d0
      ai=0.d0

      do n=1,lc
      wn=wx(n)
      t=x(n)*cs-ax(n)
      ar=ar+wn*cos(t)
      ai=ai+wn*sin(t)
      end do !n

      ab=rsq*(ar*ar+ai*ai)*bsq
      vk=vk+ab
      if (ab .lt. big) go to 10
      big=ab
      phib=phim
      thetab=k
10    end do !k

      v=v+wm*sm*dm*vk
      end do !m

      v=v*ba2*dt                        !  trapezoidal and gauss rules
      two=two*ba2
```

```
      dfw=two*2.d0*pi*w3/v
      diw=10.d0*log10(dfw)
      print 21, v
21    format("double integral",d25.16)
      print 4, dfw,diw
4     format("dfw, diw",2d25.16)
      dfb=two*2.d0*pi*big/v          !  directivity factor
      dib=10.d0*log10(dfb)           !  directivity index
      print 5, dfb,dib
5     format("dfb, dib",2d25.16)

      sec=dtime(time)
      print 3, sec,time
9     end do !j

      thetab=thetab*dt
      print 2
      print 6, big,w3                !  big = maximum power response
6     format("big, w3",2d25.16)
      print 7, phib,thetab           !  angles after coarse search
7     format("phib, thetab",2d25.16)

12    phio=phib                      !  fine search for maximum,
      thetao=thetab                  !  starting from phib,thetab
      do i=-1,1
      phi=phio+dphi*i
      cm=cos(phi)
      sm=sin(phi)
      do j=-1,1
      theta=thetao+dtheta*j
      cs=cos(theta)*sm
      ss=sin(theta)*sm
      u=1.d0-ss                      !  element response:
      rsq=exp(-2.d0*u*(1.d0+u*(.5d0+u*(t3+u*(.25d0+u*.2d0)))))
      ar=0.d0
      ai=0.d0
      br=0.d0
      bi=0.d0

      do n=1,lc
      wn=wx(n)
      t=x(n)*cs-ax(n)
      ar=ar+wn*cos(t)
      ai=ai+wn*sin(t)
      end do !n

      do n=1,nc
      wn=wz(n)
      t=z(n)*cm-az(n)
      br=br+wn*cos(t)
      bi=bi+wn*sin(t)
      end do !n

      ab=rsq*(ar*ar+ai*ai)*(br*br+bi*bi)
      if (ab .le. big) go to 11
      big=ab
      phib=phi
      thetab=theta
11    end do !j
      end do !i

      if (abs(phib-phio)+abs(thetab-thetao) .gt. 0.d0) go to 12
      print 2
      print 8, big
8     format("big",2d25.16)
      print 7, phib,thetab           !  angles after fine search
```

```
      dfb=two*2.d0*pi*big/v          !  directivity factor
      dib=10.d0*log10(dfb)           !  directivity index
      print 5, dfb,dib

      phib=phis                      !  fine search for maximum,
      thetab=thetas                  !  starting from phis,thetas
      big=0.d0
      k=0
14    phio=phib
      thetao=thetab
      do i=-1,1
      phi=phio+dphi*i
      if (k .eq. 0) phi=phis
      cm=cos(phi)
      sm=sin(phi)
      do j=-1,1
      theta=thetao+dtheta*j
      if (k .eq. 0) theta=thetas
      cs=cos(theta)*sm
      ss=sin(theta)*sm
      u=1.d0-ss                      !  element response:
      rsq=exp(-2.d0*u*(1.d0+u*(.5d0+u*(t3+u*(.25d0+u*.2d0)))))
      ar=0.d0
      ai=0.d0
      br=0.d0
      bi=0.d0

      do n=1,lc
      wn=wx(n)
      t=x(n)*cs-ax(n)
      ar=ar+wn*cos(t)
      ai=ai+wn*sin(t)
      end do !n

      do n=1,nc
      wn=wz(n)
      t=z(n)*cm-az(n)
      br=br+wn*cos(t)
      bi=bi+wn*sin(t)
      end do !n

      ab=rsq*(ar*ar+ai*ai)*(br*br+bi*bi)
      if (ab .le. big) go to 13
      big=ab
      phib=phi
      thetab=theta
      if (k .eq. 0) go to 16
13    end do !j
      end do !i
16    k=k+1
      if (k .eq. 1) go to 14

      if (abs(phib-phio)+abs(thetab-thetao) .gt. 0.d0) go to 14
      print 2
      print 8, big
      print 7, phib,thetab           !  angles near steering direction
      dfs=two*2.d0*pi*big/v          !  directivity factor
      dis=10.d0*log10(dfs)           !  directivity index
      print 15, dfs,dis
15    format("dfs, dis",2d25.16)

      end
```

APPENDIX D — Program planar-xz-equal.f

In this appendix, coding for Hamming array weighting has been included in the listed program (as an option) for either or both of the x- and z-coordinates. However, it has been exercised in only two of the numerical examples that we present here.

```
      implicit real*8(a-h,o-z)          !  planar-xz-equal.f
      real sec,dtime,time(1:2)
      parameter(lc=200,nc=50,kc=400,ng=2520) !  kc = number of theta samples
      dimension xx(1:ng),ww(1:ng),wx(1:lc),wz(1:nc),c(1:kc),s(1:kc)
      dimension mm(0:12)/0,16,24,32,48,64,96,128,192,256,384,512,768/
1     format(d28.18) ! case 1  2  3  4  5  6   7   8   9  10  11  12 <-- j
2     format(2d25.16)
3     format("sec",4e15.5)
      pi=4.d0*atan(1.d0)

      sec=dtime(time)                   !  use < legendre-array-dp
      read 1, xx(1:ng)                  !  gauss-legendre locations
      read 1, ww(1:ng)                  !  gauss-legendre weights
      sec=dtime(time)
      print 3, sec,time
      print 2

      ampd=.9d0                         !  relative power dip,  .le. 1
      phid=pi*.5d0                      !  polar angle of dip, radians
      sigd=.1d0                         !  width of dip, radians

      freq=4000.d0                      !  frequency, hertz
      speed=4900.d0*12.d0               !  sound speed, inches/second
      dx=3.5d0                          !  x spacing, inches
      dz=3.6d0                          !  z spacing, inches
      a=0.d0                            !  lower limit on phi, radians
      b=pi                              !  upper limit on phi, radians
      phis=pi*.5d0                      !  polar steering angle, radians
      thetas=pi*.5d0                    !  azimuth steering angle, radians
      dphi=.0001d0                      !  phi increment in search, radians
      dtheta=.0001d0                    !  theta increment in search, radians
      j1=8                              !  starting gauss case, j1 .ge. 1
      j2=10                             !  ending   gauss case, j2 .le. 12

      wavelength=speed/freq
      cx=2.d0*pi/wavelength*dx
      cz=2.d0*pi/wavelength*dz
      bx=cx*cos(thetas)*sin(phis)
      bz=cz*cos(phis)

      w1=0.d0
      t=(lc+1)*.5d0
      if (lc .gt. 1) go to 22
      u=0.d0
      go to 23
22    u=2.d0*pi/(lc-1)
23    do n=1,lc
c     wx(n)=.54d0+.46d0*cos(u*(n-t))    !  x element weights (Hamming)
      wx(n)=1.d0                        !  x element weights (arbitrary)
      w1=w1+wx(n)
      end do

      w2=0.d0
      t=(nc+1)*.5d0
      if (nc .gt. 1) go to 24
      u=0.d0
      go to 25
24    u=2.d0*pi/(nc-1)
25    do n=1,nc
c     wz(n)=.54d0+.46d0*cos(u*(n-t))    !  z element weights (Hamming)
      wz(n)=1.d0                        !  z element weights (arbitrary)
      w2=w2+wz(n)
      end do
      w3=w1*w1*w2*w2

      dt=2.d0*pi/kc                     !  theta sampling increment
```

```
do k=1,kc
t=k*dt                                  !  theta(k)
c(k)=cos(t)
s(k)=sin(t)
end do

ba1=(b+a)*.5d0
ba2=(b-a)*.5d0
m22=0
big=0.d0
t3=1.d0/3.d0

do j=1,12
mc=mm(j)                                !  mc = number of phi samples
m22=m22+mm(j-1)
if (j .lt. j1) go to 9
if (j .gt. j2) go to 9
v=0.d0
two=0.d0

do m=1,mc
wm=ww(m+m22)
phim=ba1+ba2*xx(m+m22)
sm=sin(phim)
fx=cx*sm
ez=cz*cos(phim)-bz
dm=(phim-phid)/sigd
dm=1.d0-ampd*exp(-.5d0*dm*dm)           !  noise power distribution
vk=0.d0
two=two+wm*sm*dm
br=0.d0
bi=0.d0

r=1.d0
q=0.d0
ce=cos(ez)
se=sin(ez)
do n=1,nc
t=r*ce-q*se
q=q*ce+r*se
r=t
wn=wz(n)
br=br+wn*r
bi=bi+wn*q
end do !n
bsq=br*br+bi*bi

do k=1,kc
u=1.d0-s(k)*sm                          !  element response:
rsq=exp(-2.d0*u*(1.d0+u*(.5d0+u*(t3+u*(.25d0+u*.2d0)))))
ex=fx*c(k)-bx
ar=0.d0
ai=0.d0

r=1.d0
q=0.d0
ce=cos(ex)
se=sin(ex)
do n=1,lc
t=r*ce-q*se
q=q*ce+r*se
r=t
wn=wx(n)
ar=ar+wn*r
ai=ai+wn*q
end do !n
```

```
        ab=rsq*(ar*ar+ai*ai)*bsq
        vk=vk+ab
        if (ab .lt. big) go to 10
        big=ab
        phib=phim
        thetab=k
10      end do !k

        v=v+wm*sm*dm*vk
        end do !m

        v=v*ba2*dt                      !  trapezoidal and gauss rules
        two=two*ba2
        dfw=two*2.d0*pi*w3/v
        diw=10.d0*log10(dfw)
        print 21, v
21      format("double integral",d25.16)
        print 4, dfw,diw
4       format("dfw, diw",2d25.16)
        dfb=two*2.d0*pi*big/v           !  directivity factor
        dib=10.d0*log10(dfb)            !  directivity index
        print 5, dfb,dib
5       format("dfb, dib",2d25.16)

        sec=dtime(time)
        print 3, sec,time
9       end do !j

        thetab=thetab*dt
        print 2
        print 6, big,w3                 !  big = maximum power response
6       format("big, w3",2d25.16)
        print 7, phib,thetab            !  angles after coarse search
7       format("phib, thetab",2d25.16)

12      phio=phib                       !  fine search for maximum,
        thetao=thetab                   !  starting from phib,thetab
        do i=-1,1
        phi=phio+dphi*i
        sm=sin(phi)
        fx=cx*sm
        ez=cz*cos(phi)-bz
        br=0.d0
        bi=0.d0

        r=1.d0
        q=0.d0
        ce=cos(ez)
        se=sin(ez)
        do n=1,nc
        t=r*ce-q*se
        q=q*ce+r*se
        r=t
        wn=wz(n)
        br=br+wn*r
        bi=bi+wn*q
        end do !n
        bsq=br*br+bi*bi

        do j=-1,1
        theta=thetao+dtheta*j
        u=1.d0-sin(theta)*sm                    !  element response:
        rsq=exp(-2.d0*u*(1.d0+u*(.5d0+u*(t3+u*(.25d0+u*.2d0)))))
        ex=fx*cos(theta)-bx
        ar=0.d0
```

```
      ai=0.d0

      r=1.d0
      q=0.d0
      ce=cos(ex)
      se=sin(ex)
      do n=1,lc
      t=r*ce-q*se
      q=q*ce+r*se
      r=t
      wn=wx(n)
      ar=ar+wn*r
      ai=ai+wn*q
      end do !n

      ab=rsq*(ar*ar+ai*ai)*bsq
      if (ab .le. big) go to 11
      big=ab
      phib=phi
      thetab=theta
11    end do !j
      end do !i

      if (abs(phib-phio)+abs(thetab-thetao) .gt. 0.d0) go to 12
      print 2
      print 8, big
8     format("big",2d25.16)
      print 7, phib,thetab            !  angles after fine search
      dfb=two*2.d0*pi*big/v           !  directivity factor
      dib=10.d0*log10(dfb)            !  directivity index
      print 5, dfb,dib

      phib=phis                       !  fine search for maximum,
      thetab=thetas                   !  starting from phis,thetas
      big=0.d0
      k=0
14    phio=phib
      thetao=thetab
      do i=-1,1
      phi=phio+dphi*i
      if (k .eq. 0) phi=phis
      sm=sin(phi)
      fx=cx*sm
      ez=cz*cos(phi)-bz
      br=0.d0
      bi=0.d0

      r=1.d0
      q=0.d0
      ce=cos(ez)
      se=sin(ez)
      do n=1,nc
      t=r*ce-q*se
      q=q*ce+r*se
      r=t
      wn=wz(n)
      br=br+wn*r
      bi=bi+wn*q
      end do !n
      bsq=br*br+bi*bi

      do j=-1,1
      theta=thetao+dtheta*j
      if (k .eq. 0) theta=thetas
      u=1.d0-sin(theta)*sm                !  element response:
      rsq=exp(-2.d0*u*(1.d0+u*(.5d0+u*(t3+u*(.25d0+u*.2d0)))))
```

```
      ex=fx*cos(theta)-bx
      ar=0.d0
      ai=0.d0

      r=1.d0
      q=0.d0
      ce=cos(ex)
      se=sin(ex)
      do n=1,lc
      t=r*ce-q*se
      q=q*ce+r*se
      r=t
      wn=wx(n)
      ar=ar+wn*r
      ai=ai+wn*q
      end do !n

      ab=rsq*(ar*ar+ai*ai)*bsq
      if (ab .le. big) go to 13
      big=ab
      phib=phi
      thetab=theta
      if (k .eq. 0) go to 16
13    end do !j
      end do !i
16    k=k+1
      if (k .eq. 1) go to 14

      if (abs(phib-phio)+abs(thetab-thetao) .gt. 0.d0) go to 14
      print 2
      print 8, big
      print 7, phib,thetab            !  angles near steering direction
      dfs=two*2.d0*pi*big/v           !  directivity factor
      dis=10.d0*log10(dfs)            !  directivity index
      print 15, dfs,dis
15    format("dfs, dis",2d25.16)

      end
```

INITIAL DISTRIBUTION LIST

Addressee	No. of Copies
NAVSEASYSCOM (PEO-USW-ASTO, CDR J. Polcari, M. Traweek)	2
Defense Technical Information Center	2
ONR (V. Simons, 331SS K. Dial, R. Varley)	3
NUWC/USRD, Orlando (Code 5916, M. Young)	1
Bolt, Beranek and Newman (N. Martin, R. Brown) Contract No. N66604-95-D-0221	2
Litton Guidance and Control Systems (S. Martin) Contract No. N66604-91-C-2841	1